Surrogate Modeling for HPC Application Iteration Times Forecasting with Network Features

Xiongxiao Xu
Illinois Institute of Technology
Chicago, IL, USA
xxu85@hawk.iit.edu

Kevin A. Brown
Argonne National Laboratory
Lemont, IL, USA
kabrown@anl.gov

Tanwi Mallick
Argonne National Laboratory
Lemont, IL, USA
tmallick@anl.gov

Xin Wang
University of Illinois Chicago
Chicago, IL, USA
xwang823@uic.edu

Elkin Cruz-Camacho
Rensselaer Polytechnic Institute
Troy, NY, USA
cruzce@rpi.edu

Robert B. Ross
Argonne National Laboratory
Lemont, IL, USA
rross@mcs.anl.gov

Christopher D. Carothers
Rensselaer Polytechnic Institute
Troy, NY, USA
chrisc@cs.rpi.edu

Zhiling Lan
University of Illinois Chicago
Chicago, IL, USA
zlan@uic.edu

Kai Shu
Illinois Institute of Technology
Chicago, IL, USA
kshu@iit.edu

ABSTRACT

Interconnect networks are the foundation for modern high performance computing (HPC) systems. Parallel discrete event simulation (PDES), serving as a cornerstone in the study of large-scale networking systems by modeling and simulating the real-world behaviors of HPC facilities, faces escalating computational complexities at an unsustainable scale. The research community is interested in building a surrogate-ready PDES framework where an accurate surrogate model can be used to forecast HPC behaviors and replace computationally expensive PDES phases. In this paper, we focus on forecasting application iteration times, the key indicator of large-scale networking performance, with network features, such as bandwidth-consumed and busy time on routers. We introduce five representative methods, including LAST, Average, ARIMA, LSTM, and the proposed framework LSTM-Feat, to forecast the iteration times of an exemplar application MILC running on a dragonfly system. By incorporating network features, LSTM-Feat can understand dependencies between network features and iteration times, thus facilitating forecasts. The experiments demonstrate the effectiveness of incorporating network features into surrogate models and the potential of surrogate models to accelerate PDES.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Networks** → **Network simulations**.

KEYWORDS

Machine Learning, Parallel Discrete Event Simulation, HPC

SIGSIM PADS '24, June 24–26, 2024, Atlanta, GA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0363-8/24/06
https://doi.org/10.1145/3615979.3656055

ACM Reference Format:
Xiongxiao Xu, Kevin A. Brown, Tanwi Mallick, Xin Wang, Elkin Cruz-Camacho, Robert B. Ross, Christopher D. Carothers, Zhiling Lan, and Kai Shu. 2024. Surrogate Modeling for HPC Application Iteration Times Forecasting with Network Features. In *38th ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM PADS '24), June 24–26, 2024, Atlanta, GA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3615979.3656055

1 INTRODUCTION

The computing power of High-performance computing (HPC) systems makes them critical for a variety of applications, such as climate modeling, molecular dynamics and drug discovery. The performance of HPC systems depends on the effectiveness and scalability of their interconnect networks. One noteworthy example is the dragonfly network topology, a hierarchical, high-radix, low-diameter architecture which is able to incur reduced network cost while achieving high-bandwidth and low-latency performance [15, 16]. This topology has gained widespread adoption across various HPC facilities, e.g., the National Energy Research Scientific Computing Center and the Argonne Leadership Computing Facility.

Parallel discrete event simulation (PDES) is a computational technique used to simulate dynamic behaviors of complex systems, including internet and cybersecurity simulations, transportation and mobility applications, and hardware co-design simulations [24]. Despite the success of PDES modeling frameworks such as ROSS [4] and CODES [23], the simulation requirements and computational complexity are growing at an intractable rate. For instance, PDES requires four hours to simulate behaviors of a 4,096-node system over a 12 millisecond period [26].

To address the computational issue, an active research topic is to construct fast surrogate models to forecast specific activities and accelerate PDES. However, there still remain several unsolved challenges. One such challenge is accurately and efficiently forecasting workload-level activities, e.g., application iteration times, using data available from fine-grain simulations. For applications that work by iteratively looping over a set of recurring computing and communication tasks, such as the tasks required for daily weather predictions, the time to complete each iteration can vary significantly when network congestion delays communication operations. Furthermore, network features such as router port bandwidth and busy time can expose the state of the network congestion, but it is not straightforward how such features can be leveraged to facilitate iteration times forecasting.

To tackle the above challenges, we explore several representative surrogate modeling approaches, including LAST, Average, ARIMA, LSTM, and LSTM-Feat, for application iteration times forecasting. By taking network features into account, the proposed LSTM-Feat is able to leverage dependencies between network features and iteration times. The experiments show that incorporating network features into a surrogate model is beneficial to forecasts and promising surrogate modeling approaches to accelerate PDES. We envision a surrogate-ready PDES that seamlessly shifts between a detailed simulation of the application workload and a fast-forward surrogate phase based on forecasts of surrogate models.

2 RELATED WORK

2.1 ML for Time Series Forecasting

Machine learning (ML) for time-series forecasting has been studied for a long time [9]. The task aims to forecast a period of future data given a sequence of historical data and has a variety of applications, including transportation, finance and medicine. The earlier researchers leverage the statistical methods, e.g., ARIMA (AutoregRessive Integrated Moving Average) [3, 18], and traditional machine learning methods, such as SVM (Support Vector Machine) [5, 14] to forecast time-series data. However, they may not achieve desirable performance due to their linear assumptions. Recently, time-series forecasting has made a significant progress due to the emergence of the deep learning. Deep learning models deliver expressive performance because they can capture complex pattern in time-series data, like Convolutional Neural Network (CNN) [29, 34, 36], Recurrent Neural Network (RNN) [7, 22, 29], LSTM [1, 21, 34], Graph neural network (GNN) [6, 30, 37], Transformer [20, 25, 28] and State Space Models [2, 10, 32]. However, none of the above work aim at forecasting application iteration times in the HPC system. This paper is focused on forecasting application iteration times in the dragonfly system.

2.2 Accelerate PDES with Surrogate Models

Various multi-resolution and hybrid PDES models [11, 12, 17, 19] have been proposed to accelerate high-fidelity PDES simulations and have shown promising results. With the emergence of machine learning, the community is interested in designing [8, 31, 33] a high-fidelity ML-based surrogate model to accelerate PDES by forecasting port-level network traffic in the dragonfly system. Different from the existing work focusing on port-level network traffic forecasting, we forecast application iteration times, a workload-level characteristics, and take network features into account to facilitate forecasting accuracy with machine learning surrogate models.

3 METHODOLOGY

3.1 Background

We focus on the 1D dragonfly network as shown in Figure 1. The dragonfly network [16] has a hierarchical architecture with three

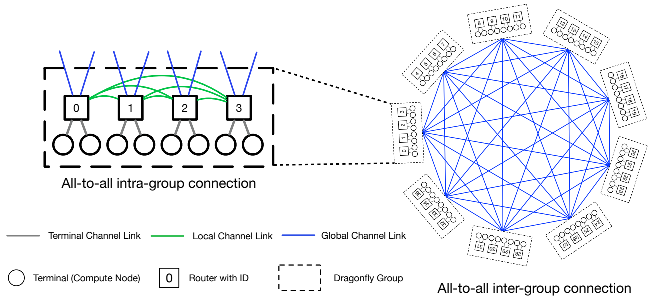

Figure 1: The illustration of the 1D Dragonfly network.

levels: router, group, and system. The system are divided into multiple groups, and a group has multiple routers. A router connects to computer nodes (terminal) by terminal channel links, connects to other routers within a group by local channel links and connects to other routers outside the group by global channel links. The connection points between routers and links are ports on routers. When running on the dragonfly network, applications are divided into multiple processes, which are then placed on the computer nodes. Each process occupies a computer nodes and the processes collaborate with each other on executing applications. The execution commonly consists of multiple steps, so each process has multiple iterations to complete. Each process is assigned a unique identifier known as a "rank"[1]. We aim to forecast application iteration times for each rank, which refers to the time it takes to complete one iteration. Iteration denotes the repetitive execution of a set of computational and communication tasks. The processes communicate with each other by sending messages, and routers forward the messages through the ports. During the communication, the ports have key characteristics in the dragonfly network, e.g., bandwidth-consumed, busy time, etc. The system characteristics reflect the status of the dragonfly network and can be potentially used to improve the accuracy of forecasts.

3.2 Problem Definition

Let the dragonfly network have n_r routers and n_c computer nodes and n_p processes placed on the computer nodes for execution. In the execution, the n_p processes work together to complete T iterations. For a rank $p \in \{1, 2, ..., n_p\}$ at an iteration $t \in \{1, 2, ..., T\}$, we use $y_{p,t}$ to denote its application iteration time, and use $x_{p,t}$ to denote network features of a router connecting to a compute node where the rank p exists. For ease of expression, we ignore the subscript p in the following descriptions and formally define a problem:

Problem Statement. Given a application iteration times and network features sequence for a rank p with look-back window $\mathcal{B} = \{y_{t-(L_x-1)}, y_{t-(L_x-2)}, ..., y_t; x_{t-(L_x-1)}, x_{t-(L_x-2)}, ..., x_t\}$ with length L_x, we aim to forecast a sequence of future application iteration times $\mathcal{F} = \{y_{t+1}, y_{t+2}, ..., y_{t+L_y}\}$ with length L_y.

3.3 Surrogate Modeling

We investigate four surrogate modeling approaches to forecast iteration times in a distributed setting where each compute node holding a rank has a surrogate model. The details are as follows:

[1]We might use process and rank interchangeable in this paper.

Table 1: Performance and inference time overhead comparison results on all ranks. I.T.O. means inference time overhead. The lower MSE and MAE denote more effective, and the lower I.T.O. denotes more efficient. The best results are in bold.

Methods	*cont-adp*			*rand-adp*		
	MSE	MAE	I.T.O. (s)	MSE	MAE	I.T.O. (s)
LAST	0.0034±0.0000	0.0291±0.0000	**0.0057±0.0017**	0.0026±0.0000	0.0176±0.0000	**0.0034±0.0001**
Average	0.0048±0.0000	0.0309±0.0000	0.0926±0.0013	0.0110±0.0000	0.0343±0.0000	0.0680±0.0005
ARIMA	0.0047±0.0000	0.0330±0.0000	15120.1573±34.715	0.0107±0.0000	0.0315±0.0000	11473.6628±19.4696
LSTM	0.0032±0.0001	0.0265±0.0004	5.8233±0.1117	0.0020±0.0001	0.0170±0.0004	4.6101±0.0371
LSTM-Feat	**0.0025±0.0001**	**0.0253±0.0016**	6.1617±0.0107	**0.0013±0.0001**	**0.0167±0.0007**	4.5913±0.0244

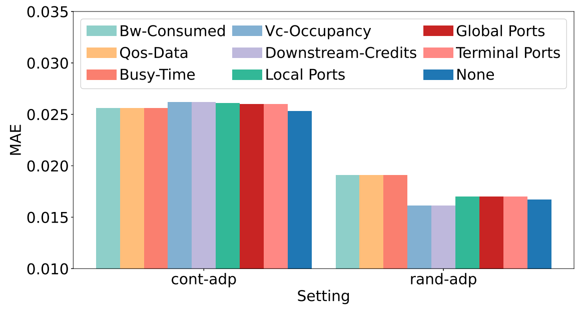

Figure 2: Feature importance analysis. The legend name denotes removing corresponding features in LSTM-Feat.

LAST is a heuristic method and forecasts the application iteration time of a future iteration as the latest historical application iteration time, i.e., $y_{t+1} = y_t$. Note that the length of the foretasted sequence and the look-back window both have to be 1, i.e., $L_x = L_y = 1$.
Average is also a heuristic method and forecasts application iteration time of a future iteration as the average of the look-back window $\mathcal{B}$, i.e., $y_{t+1} = \frac{\sum_{i=t-(L_x-1)}^{t} y_i}{L_x}$. Note that we have settled for the length of the foretasted sequence to be 1 and a length of the look-back window of 10, i.e. $L_x = 10$ and $L_y = 1$.
ARIMA (AutoRegressive Integrated Moving Average) [3] is a classical statistical time series analysis method and can handle non-stationary time series. The look-back window L_x is 250 due to the frequent occurrence of matrix decomposition errors for small L_x, i.e. $L_x = 250$. The forecasted sequence has size of 1, i.e. $L_y = 1$.
LSTM (Long Short-Term Memory) [13] is a well-known machine learning model and widely used in time series data. LSTM is effective in addressing the gradient vanishing issue in sequential modeling problems. We set $L_x = 10$ and $L_y = 1$.
LSTM-Feat is a variant of LSTM. Different from the above surrogate modeling methods where the input and output are both application iteration times, it leverages a combination of application iteration times sequences and network features sequences to forecast iteration times sequences. The motivation is to leverage the potential correlation among network features and application iteration times. We set $L_x = 10$ and $L_y = 1$.

4 EXPERIMENTS

4.1 Experimental Setting

Network Topology. The dragonfly network (see Figure 1) has a hierarchical design, consisting of the all-to-all inter-group connection and intra-group connection. Our network has 72 compute nodes and 36 routers equally divided across 9 groups. Each router has 7 ports: 2 terminal ports, 3 local ports, and 2 global ports.
Network Simulator. We utilize CODES [23] to simulate our workload. Times are collected for 2000 iterations along with network features from each port collected at 250 μs.
Network Features. The network features consist of bw-consumed, qos-data, busy-time, vc-occupancy, and downstream-credits. Qos-data is the amount of data sent by the port; bw-consumed is the percentage of the consumed bandwidth; busy-time is the total time the port was stalled, i.e. chunks were blocked from sending due to flow control; vc-occupancy is the number of bytes in each VoQ buffer of the port at the point in time when the measurement was taken; downstream-credits is the number of credits available for the respective downstream virtual channels at the point in time when the measurement was taken.
Align Datasets. We align application iteration times and network features datasets to resolve the inconsistency between them. Iteration times are on compute nodes while network features are on routers; an iteration time is recorded when an iteration completes while network features are collected at a fixed time interval. To eliminate the inconsistency, we combine the iteration time on a compute node with network features on a router connecting to the compute node; we search time points in network features dataset that are the nearest to iteration times, and take the network features of the time points to combine the iteration times.
HPC Workload. The workload includes: (1) *MILC* is a HPC application used to study quantum chromodynamics (QCD) and features numerous nonblocking send/receive communication operations. (2) *UR* is a synthetic traffic featuring each node sending successive messages to a random destination. The messages are streamed at user-defined injection loads, alternating between 10% and 100%.
Job Placement. We investigate two job placements: (1) *Contiguous Placement* selects computer nodes consecutively for the processes of the job to occupy. (2) *Random Placement* selects computer nodes randomly for the processes of the job to occupy.
Routing Policy. *Progressive adaptive routing* [27] is used in our simulation. Packets are routed along minimal or non-minimal paths based on the network congestion state. When a non-minimal path is selected, the packet will be minimally routed into a randomly intermediate router, and then minimally forwarded to its destination. According to job placement and routing policy strategies, we denote the two settings as *cont-adp* and *rand-adp*, respectively.
Evaluation Metrics. We use metrics **MSE** (Mean Square Error) and **MAE** (Mean Absolute Error) to assess the effectiveness of

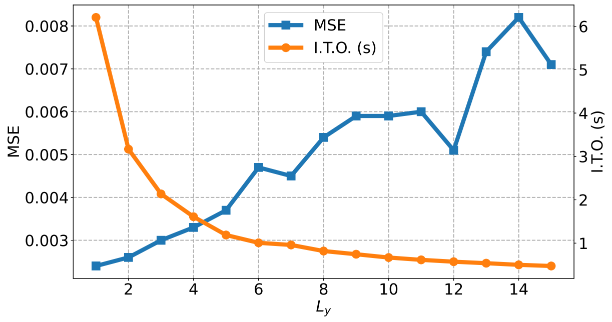

Figure 3: The impact of the length L_y on the MSE and inference time overhead in LSTM-FEAT.

models [34, 35], and **I.T.O.** (Inference Time Overhead) measures time overhead when the surrogate models do forecasts.

Implementation Details. We split 2000 iterations into training, validation, and test data. The split ratio is 6:2:2 for *cont-adp* and 7:1.5:1.5 for *rand-adp* due to their different distributions. We normalize iteration times into range $[0, 1]$ for stability of training. We run experiments 3 times and report the average and standard deviation.

4.2 Experimental Results

Application Iteration Times Forecasting. We show the performance comparison for the foreasts in Table 1. According to the experimental results, we have the following observations:

- In terms of MSE and MAE, the LSTM-FEAT outperforms other surrogate models, including heuristical methods, i.e., LAST and Average, traditional statistical method, i.e., ARIMA, and deep learning methods without considering network features, i.e., LSTM. For instance, LSTM-FEAT outperforms LAST and LSTM by 50% and 35%, respectively, w.r.t. MSE. It demonstrates LSTM-FEAT can capture temporal patterns in iteration times sequences and dependencies between network features and iteration times.
- In terms of I.T.O., LAST and Average are the two fastest methods as the implementation of the heuristic methods are simple. ARIMA is the slowest method as such a statistical method needs to calculate parameters by fitting a set of data in a long look-back window per step. The efficiency of deep learning methods are between the above two kind of methods. For exmaple, LSTM-FEAT is slower than LAST but faster than ARIMA. However, we show deep learning is potential w.r.t. I.T.O. in the sensitivity analysis.

Feature Importance Analysis. We conduct feature importance analysis in Figure 2. In detail, we remove a specific network feature across all ports of a router, e.g., bw-consumed or busy-time, and remove all features from a type of ports of a router, e.g.., local or global ports, to observe the performance change of the LSTM-FEAT. Accordingly, we observe the following points:

- Network features are generally helpful to iteration times forecasting. For example, in the *rand-adp* configuration, if we remove bw-consumed feature, MAE increases from 0.0167 to 0.0191. It shows dependencies exist between bw-consumed and iteration times. On the other hand, we also note that vc-occupancy and downstream-credits are useful in *cont-adp* configuration while they cannot help forecast iteration times in *rand-adp* configuration. It may be because both features capture the state of port buffers for a single point in time during an iteration unlike other features that capture the behavior of the port throughout the iteration. Furthermore, adaptive routing causes buffer occupancies to vary somewhat stochastically, making it difficult to associate occupancy trends to specific per-rank application activities.
- Network features on all types of ports of a router are important to forecast application iteration times. For instance, in the *cont-adp* configuration, MAE of LSTM-FEAT increases from 0.0253 to 0.0261 with the removal of network features on local ports.

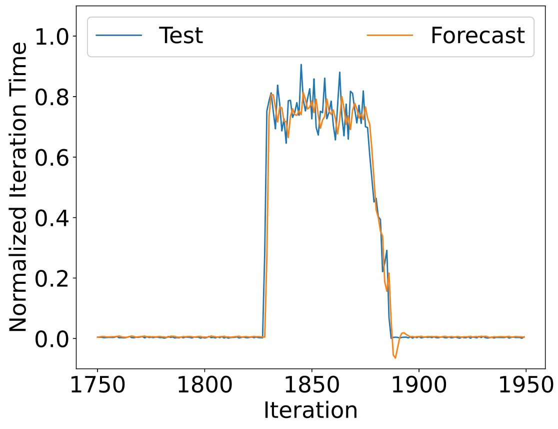

Figure 4: The forecast visualization for rank 0 of LSTM-FEAT.

Sensitivity Analysis. We investigate the impact of the length L_y of the future values sequence $\mathcal{F}$ on LSTM-FEAT. Particularly, we vary the L_y and record values of metrics as shown in Figure 3. We observe there is a trade-off between effectiveness (MSE) and efficiency (I.T.O.) for LSTM-FEAT. When the length L_y increases, MSE generally increases but I.T.O. consistently decreases. It means the effectiveness of LSTM-FEAT decreases but the efficiency increases with the L_y increasing. For effectiveness diminishing, the reason is that forecasting farther length at a time is more difficult due to more uncertainty in the father length. With regard to efficiency increasing, it is because the required inference times are reduced if increasing forecasting length at a time given the fixed total forecasting length. For instance, if the total forecasting length is 200 and the L_y is 1, the required inference times are 200; if the L_y is 10, the required inference times are reduced to 20. It shed light to a promising direction where deep learning methods can achieve both satisfactory effectiveness and efficiency if choosing appropriate L_y.

Visualization. The visualization of a rank is shown in Figure 4. LSTM-FEAT can achieve satisfactory forecasting performance.

5 CONCLUSION AND FUTURE WORK

In this paper, we investigate surrogate modeling approaches to forecast application iteration times for computational challenges in PDES. Our results demonstrate the superiority of LSTM-FEAT as a surrogate model and potential to incorporate network features. The future work may include: (1) decreasing the time overhead of deep learning methods, and (2) improving long-term forecast accuracy as the length of future value sequence L_y grows.

Acknowledgement: This work is supported by the Director, Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-06CH11357.

REFERENCES

[1] Hossein Abbasimehr and Reza Paki. 2022. Improving time series forecasting using LSTM and attention models. *Journal of Ambient Intelligence and Humanized Computing* 13, 1 (2022), 673–691.

[2] Md Atik Ahamed and Qiang Cheng. 2024. TimeMachine: A Time Series is Worth 4 Mambas for Long-term Forecasting. *arXiv preprint arXiv:2403.09898* (2024).

[3] Mohammed S Ahmed and Allen R Cook. 1979. *Analysis of freeway traffic time-series data by using Box-Jenkins techniques*. Number 722.

[4] Christopher D Carothers, David Bauer, and Shawn Pearce. 2002. ROSS: A high-performance, low-memory, modular Time Warp system. *J. Parallel and Distrib. Comput.* 62, 11 (2002), 1648–1669.

[5] Manoel Castro-Neto, Young-Seon Jeong, Myong-Kee Jeong, and Lee D Han. 2009. Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert systems with applications* 36, 3 (2009), 6164–6173.

[6] Di Chai, Leye Wang, and Qiang Yang. 2018. Bike flow prediction with multi-graph convolutional networks. In *Proceedings of the 26th ACM SIGSPATIAL international conference on advances in geographic information systems*. 397–400.

[7] Xingyi Cheng, Ruiqing Zhang, Jie Zhou, and Wei Xu. 2018. Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting. In *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[8] Elkin Cruz-Camacho, Kevin A Brown, Xin Wang, Xiongxiao Xu, Kai Shu, Zhiling Lan, Robert B Ross, and Christopher D Carothers. 2023. Hybrid PDES Simulation of HPC Networks Using Zombie Packets. In *Proceedings of the 2023 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (Orlando, FL, USA)(SIGSIM-PADS'23). Association for Computing Machinery, New York, NY, USA.*

[9] Jan G De Gooijer and Rob J Hyndman. 2006. 25 years of time series forecasting. *International journal of forecasting* 22, 3 (2006), 443–473.

[10] Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752* (2023).

[11] Yu Gu, Yong Liu, and Don Towsley. 2004. On integrating fluid models with packet simulation. In *IEEE INFOCOM 2004*, Vol. 4. IEEE, 2856–2866.

[12] Qi He, Mostafa Ammar, George Riley, and Richard Fujimoto. 2002. Exploiting the predictability of TCP's steady-state behavior to speed up network simulation. In *Proceedings. 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*. IEEE, 101–108.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Xuexiang Jin, Yi Zhang, and Danya Yao. 2007. Simultaneously prediction of network traffic flow based on PCA-SVR. In *Advances in Neural Networks–ISNN 2007: 4th International Symposium on Neural Networks, ISNN 2007, Nanjing, China, June 3-7, 2007, Proceedings, Part II 4*. Springer, 1022–1031.

[15] Yao Kang, Xin Wang, and Zhiling Lan. 2022. Study of Workload Interference with Intelligent Routing on Dragonfly. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis* (Dallas, Texas) *(SC '22)*. Article 20, 14 pages.

[16] John Kim, Wiliam J Dally, Steve Scott, and Dennis Abts. 2008. Technology-driven, highly-scalable dragonfly topology. *ACM SIGARCH Computer Architecture News* 36, 3 (2008), 77–88.

[17] Patrick Lavin, Jeffrey Young, and Richard Vuduc. [n.d.]. Multifidelity Memory System Simulation in SST. ([n. d.]).

[18] Moshe Levin and Yen-Der Tsao. 1980. On forecasting freeway occupancies and volumes (abridgment). *Transportation Research Record* 773 (1980).

[19] Jason Liu. 2007. Parallel simulation of hybrid network traffic models. In *21st International Workshop on Principles of Advanced and Distributed Simulation (PADS'07)*. IEEE, 141–151.

[20] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2023. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625* (2023).

[21] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang. 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies* 54 (2015), 187–197.

[22] Rishabh Madan and Partha Sarathi Mangipudi. 2018. Predicting computer network traffic: a time series forecasting approach using DWT, ARIMA and RNN. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*. IEEE, 1–5.

[23] Misbah Mubarak, Christopher D Carothers, Robert B Ross, and Philip Carns. 2017. Enabling parallel simulation of large-scale HPC network systems. *IEEE Transactions on Parallel and Distributed Systems* 28, 1 (2017), 87–100.

[24] Kalyan Perumalla, Maximilian Bremer, Kevin Brown, Cy Chan, Stephan Eidenbenz, K Scott Hemmert, Adolfy Hoisie, Benjamin Newton, James Nutaro, Tomas Oppelstrup, et al. 2022. *Computer Science Research Needs for Parallel Discrete Event Simulation (PDES).* Technical Report. Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States).

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[26] Xin Wang, Misbah Mubarak, Yao Kang, Robert B Ross, and Zhiling Lan. 2020. Union: An automatic workload manager for accelerating network simulation. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 821–830.

[27] Xin Wang, Misbah Mubarak, Xu Yang, Robert B Ross, and Zhiling Lan. 2018. Trade-off study of localizing communication and balancing network traffic on a dragonfly system. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 1113–1122.

[28] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2022. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125* (2022).

[29] Yuankai Wu, Huachun Tan, Lingqiao Qin, Bin Ran, and Zhuxi Jiang. 2018. A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies* 90 (2018), 166–180.

[30] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 753–763.

[31] Xiongxiao Xu. 2023. Exploring Machine Learning Models with Spatial-Temporal Information for Interconnect Network Traffic Forecasting. In *Proceedings of the 2023 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. 56–57.

[32] Xiongxiao Xu, Yueqing Liang, Baixiang Huang, Zhiling Lan, and Kai Shu. 2024. Integrating Mamba and Transformer for Long-Short Range Time Series Forecasting. *arXiv preprint arXiv:2404.14757* (2024).

[33] Xiongxiao Xu, Xin Wang, Elkin Cruz-Camacho, Christopher D. Carothers, Kevin A. Brown, Robert B. Ross, Zhiling Lan, and Kai Shu. 2023. Machine Learning for Interconnect Network Traffic Forecasting: Investigation and Exploitation. In *Proceedings of the 2023 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. 133–137.

[34] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5668–5675.

[35] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2022. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504* (2022).

[36] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-first AAAI conference on artificial intelligence*.

[37] Jiawei Zhu, Xing Han, Hanhan Deng, Chao Tao, Ling Zhao, Pu Wang, Tao Lin, and Haifeng Li. 2022. KST-GCN: A knowledge-driven spatial-temporal graph convolutional network for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems* 23, 9 (2022), 15055–15065.